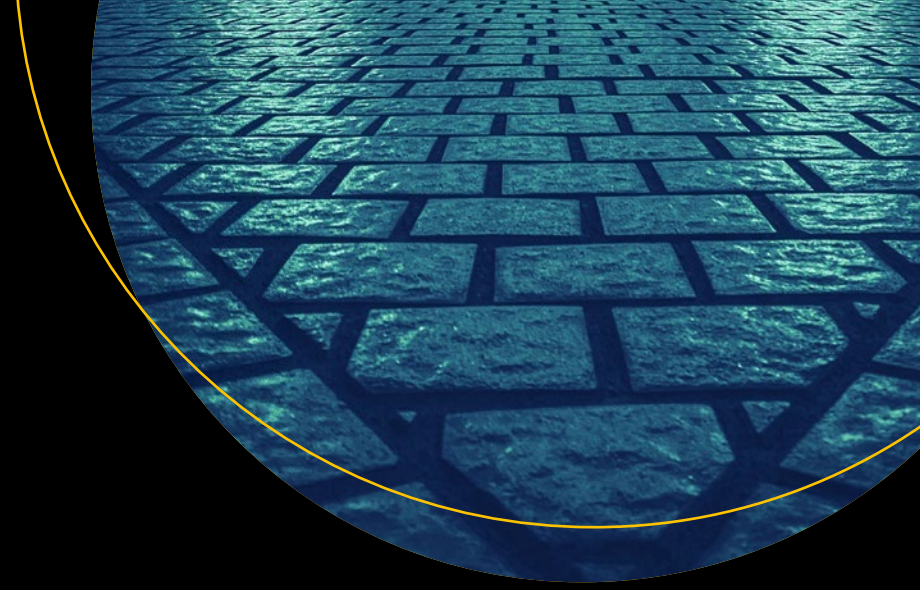

Options and Derivatives Programming in C++20

Algorithms and Programming Techniques for the Financial Industry

—

Second Edition

—

Carlos Oliveira

Apress®

Options and Derivatives Programming in C++20

Algorithms and Programming Techniques for the Financial Industry

Second Edition

Carlos Oliveira

Apress®

Options and Derivatives Programming in C++20: Algorithms and Programming Techniques for the Financial Industry

Carlos Oliveira
Seattle, WA, USA

ISBN-13 (pbk): 978-1-4842-6314-3 ISBN-13 (electronic): 978-1-4842-6315-0
https://doi.org/10.1007/978-1-4842-6315-0

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Steve Anglin
Development Editor: Matthew Moodie
Coordinating Editor: Mark Powers

Cover designed by eStudioCalamar

Cover image by Peixin Wu on Unsplash (www.unsplash.com)

Distributed to the book trade worldwide by Apress Media, LLC, 1 New York Plaza, New York, NY 10004, U.S.A. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail booktranslations@springernature.com; for reprint, paperback, or audio rights, please e-mail bookpermissions@springernature.com.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at http://www.apress.com/bulk-sales.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book's product page, located at www.apress.com/9781484263143. For more detailed information, please visit http://www.apress.com/source-code.

Printed on acid-free paper

To my family, my real source of inspiration.

Table of Contents

About the Author

Carlos Oliveira works in the area of quantitative finance, with more than 15 years of experience in creating scientific and financial models in C++. During his career, Carlos has developed several large-scale applications for financial companies such as Bloomberg L.P. and Incapital LLC. Carlos obtained a PhD in Operations Research and Systems Engineering from the University of Florida, an MSc in Computer Science from UFC (Brazil), and a BSc in Computer Science from UECE (Brazil). He also performs academic research in the field of combinatorial optimization, with applications in diverse areas such as finance, telecommunications, computational biology, transportation, and logistics. Carlos Oliveira has written more than 30 academic papers on optimization and authored four books, including *Practical C++ Financial Programming* (Apress, 2015).

About the Technical Reviewer

David Pazmino has been developing software applications for 20 years in Fortune 100 companies. He is an experienced developer in front-end and back-end development who specializes in developing machine learning models for financial applications. David has developed many applications in C++, STL, and ATL for companies using Microsoft technologies. He currently develops applications in Scala and Python for deep learning neural networks. David has a degree from Cornell University, a masters from Pace University in computer science, and a masters from Northwestern University in predictive analytics.

Introduction

On Wall Street, the use of algorithmic trading and other computational techniques has skyrocketed in the last few years, as can be seen from the public interest in automated trading as well as the profits generated by these strategies. This growing trend demonstrates the importance of using software to analyze and trade markets in diverse areas of finance. One particular area that has been growing in importance during the last decade is options and derivatives trading.

Initially considered only as a niche investment strategy, derivatives have become one of the most common instruments for investors in all areas. Likewise, the interest in automated trading and analysis of such instruments has also increased considerably.

Along with scientists and economists, software engineers have greatly contributed to the development of advanced computational techniques using financial derivatives. Such techniques have been used at banks, hedge funds, pension funds, and other financial institutions. In fact, every day new systems are developed to give a trading advantage to the players in this industry.

This books attempts to provide the basic programming knowledge needed by C++ programmers working with options and derivatives in the financial industry. This is a hands-on book for programmers who want to learn how C++ is used to develop solutions for options and derivatives trading. In the book's chapters, you'll explore the main algorithms and programming techniques used in the implementation of systems and solutions for trading options and other derivatives.

Because of stringent performance characteristics, most of these trading systems are developed using C++ as the main implementation language. This makes the topic of this book relevant to everyone interested in acquiring the programming skills necessary in the financial industry.

In *Options and Derivatives Programming in C++20*, I cover the features of the language that are more frequently used to write financial software for options and derivatives. These features include the STL, templates, functional programming, and support for numerical libraries. New features introduced in the latest updates of the C++ standard are also covered, including additional functional techniques such as lambda functions, automatic type detection, custom literals, and improved initialization strategies for C++ objects.

I also provide how-to examples that cover the major tools and concepts used to build working solutions for quantitative finance. The book teaches you how to employ advanced C++ concepts as well as the basic building libraries used by modern C++ developers, such as the STL, Boost, and QuantLib. I also discuss how to create correct and efficient applications, leveraging knowledge of object-oriented and template-based programming. I assume only a basic knowledge of C and C++. Throughout this book, a number of more advanced concepts, already mastered by experienced developers, will be introduced as needed.

In the process of writing this book, I was also concerned with providing value for readers who are trying to use their current programming knowledge in order to become proficient at the style of programming used in large banks, hedge funds, and other investment institutions. Therefore, the topics covered in the book are introduced in a logical and structured way. Even novice programmers will be able to absorb the most important topics and competencies necessary to develop in C++ for the problems occurring on the analysis of options and other financial derivatives.

In this book, we also discuss features introduced in the latest international standard, C++20. In this version of the standard, a number of simplifications and extensions of the core C++ language have been approved. You will learn about many of the features in the new standard, with examples to illustrate each major concept. An appendix has been included, with detailed information about standard features added to C++ during the last decade.

Audience

This book is intended for readers who already have a working knowledge of programing in C, C++, or another mainstream language. These are usually professionals or advanced students in computer science, engineering, and mathematics, who have interest in learning about options and derivatives programming using the C++ language, for personal or for professional reasons. The book is also directed at practitioners of C++ programming in financial institutions, who would use the book as a ready-to-use reference of software development algorithms and best practices for this important area of finance.

Many readers are interested in a book that would describe how modern C++ techniques are used to solve practical problems arising when considering options on financial instruments and other derivatives. Being a multi-paradigm language, C++ usage may be slightly different in each area, so the skills that are useful for developing desktop applications, for example, are not necessarily the same ones used to write high-performance software.

A large part of high-performance financial applications are written in C++, which means that programmers who want to enter this lucrative market need to acquire a working knowledge of specific parts of the language. This book attempts to give developers who want to develop their knowledge effectively this choice, while learning one of the most sought-after and marketable skillsets for modern financial application and high-performance software development.

This book is also targeted at students and new developers who have some experience with the C++ language and want to leverage that knowledge into financial software development. This book is written with the goal of reaching readers who need a concise, algorithms-based strategy, providing basic information through well-targeted examples and ready-to-use solutions. Readers will be able to directly apply the concepts and sample code to some of the most common problems faced regarding the analysis of options and derivative contracts.

What You Will Learn

Here is a sample of topics that are covered in the following chapters:

- Fundamental problems in the options and derivatives market
  - Options market models
  - Derivative valuation problems
  - Trading strategies for options and derivatives
- Pricing algorithms for derivatives
  - Binomial method
  - Differential equations method
  - Black-Scholes model

- Quantitative finance algorithms for options and derivatives
  - Linear algebra techniques
  - Interpolation
  - Calculating roots
  - Numerical solution for PDEs
- Important features of C++ language as used in quantitative financial programming, such as
  - Templates
  - STL containers
  - STL algorithms
  - Boost libraries
  - QuantLib
  - New features of C++20

Book Contents

Here is a quick overview of the major topics covered in each chapter:

- Chapter 1—"Options Concepts": An option is a standard financial contract that derives its value from an underlying asset such as a stock. Options can be used to pursue multiple economic objectives, such as hedging against variations on the underlying asset, or speculating on the future price of a stock. Chapter 1 presents the basic concepts of options, including their advantages and challenges. It also explains how options can be modeled using C++. The main topics covered in this chapter are as follows:
  - Basic definitions of options
  - An introduction to options strategies
  - Describing options with Greeks
  - Sample code for options handling

- Chapter 2—"Financial Derivatives": A derivative is a general term for a contract whose price is based on an underlying asset. In the previous decades, the financial industry created and popularized a large number of derivatives. Pricing and trading these derivatives is a large part of the work performed by trading desks throughout the world. Derivatives have been created based on diverse assets such as foreign currency, mortgage contracts, and credit default swaps. This chapter explores this type of financial instrument and presents a few C++ techniques to model specific derivatives. The main topics covered in this chapter are as follows:
  - Credit default swaps
  - Forex derivatives
  - Interest rate derivatives
  - Exotic derivatives
- Chapter 3—"Basic C++ Algorithms": To become a proficient C++ developer, it is essential to have good knowledge of the basic algorithms used in your application area. Some basic algorithms for tasks such as vector processing, date and time handling, and data access and storage are useful in almost all applications involving options and other financial derivatives. This chapter surveys such algorithms and their implementation in C++, including the following topics:
  - Date and time handling
  - Vector processing
  - Graphs and networks
  - Fast data processing
- Chapter 4—"Object-Oriented Techniques": For the last 30 years, object-oriented techniques have become the standard for software development. Since C++ fully supports OO programming, it is imperative that you have a good understanding of OO techniques in order to solve the problems presented by options and derivatives.

I present a summary of what you need to become proficient in the relevant OO techniques used in the financial industry. Some of the topics covered in this chapter are as follows:

- Problem partitioning
- Designing solutions using OO strategies
- OO implementation in C++
- Reusing OO components

- Chapter 5—"Design Patterns for Options Processing": Design patterns are a set of common programming practices that can be used to simplify the solution of recurring problems. With the use of OO techniques, design patterns can be cleanly implemented as a set of classes that interact toward the solution of a common goal. In this chapter, you learn about the most common design patterns employed when working with financial options and derivatives, with specific examples. It covers the following topics:
  - The importance of design patterns
  - Factory pattern
  - Visitor pattern
  - Singleton pattern
  - Less common patterns
- Chapter 6—"Template-Based Techniques": C++ templates allow programmers to write code that works without modification on different data types. Through the careful use of templates, C++ programmers can write code with high performance and low overhead, without the need to employ more computationally expensive object-oriented techniques. This chapter explores a few template-oriented practices used in the solution of options- and derivatives-based financial problems:
  - Motivating the use of templates
  - Compile-time algorithms

    - Containers and smart pointers
    - Template libraries
- Chapter 7—"STL for Derivatives Programming": Modern financial programming in C++ makes heavy use of template-based algorithms. Many of the basic template algorithms are implemented in the standard template library (STL). This chapter discusses the STL and how it can be used in quantitative finance projects, in particular to solve options and financial derivative problems. You will get a clear understanding of how the STL interacts with other parts of the system, and how it imposes a certain structure on classes developed in C++.
    - STL-based algorithms
    - Functional techniques on STL
    - STL containers
    - Smart pointers
- Chapter 8—"Functional Programming Techniques": Functional programming is a technique that focuses on the direct use of functions as first-class objects. This means that you are allowed to create, store, and call functions as if they were just another variable of the system. Recently, functional techniques in C++ have been greatly improved with the adoption of the new standard (C++20), particularly with the introduction of lambda functions. The following topics are explored in this chapter:
    - Lambdas
    - Functional templates
    - Functions as first-class objects
    - Managing state in functional programming
    - Functional techniques for options processing

- Chapter 9—"Linear Algebra Algorithms": Linear algebra techniques are used throughout the area of financial engineering and, in particular, in the analysis of options and other financial derivatives. Therefore, it is important to understand how the traditional methods of linear algebra can be applied in C++. With this goal in mind, I present a few examples that illustrate how to use some of the most common linear algebra algorithms. In this chapter, you will also learn how to integrate existing LA libraries into your code.
    - Implementing matrices
    - Matrix decomposition
    - Computing determinants
    - Solving linear systems of equations
- Chapter 10—"Algorithms for Numerical Analysis": Equations are some of the building blocks of financial algorithms for options and financial derivatives, and it is important to be able to efficiently calculate the solution for such mathematical models. In this chapter, you will see programming recipes for different methods of calculating equation roots and integrating functions, along with explanations of how they work and when they should be used. I also discuss numerical error and stability issues that present a challenge for developers in the area of quantitative financial programming.
    - Basic numerical algorithms
    - Root-finding algorithms
    - Integration algorithms
    - Reducing errors in numerical algorithms
- Chapter 11—"Models Based on Differential Equations": Differential equations are at the heart of many techniques used in the analysis of derivatives. There are several processes for solving and analyzing PDEs that can be implemented in C++. This chapter presents programming recipes that cover aspects of PDE-based option modeling and application in C++. Topics covered include the following:

  - Basic techniques for differential equations
  - Ordinary differential equations
  - Partial differential equations
  - Numerical algorithms for differential equations
- Chapter 12—"Basic Models for Options Pricing": Options pricing is the task of determining the fair value of a particular option, given a set of parameters that exactly determine the option type. This chapter discusses some of the most popular models for options pricing. They include tree-based methods, such as binomial and trinomial trees. This chapter also discusses the famous Black-Scholes model, which is frequently used as the basis for the analysis of most options and derivative contracts.
  - Binomial trees
  - Trinomial trees
  - Black-Scholes model
  - Implementation strategies
- Chapter 13—"Monte Carlo Methods": Among other programming techniques for equity markets, Monte Carlo simulation has a special place due to its wide applicability and easy implementation. These methods can be used to forecast prices or to validate options buying strategies, for example. This chapter provides programming recipes that can be used as part of simulation-based algorithms applied to options pricing.
  - Probability distributions
  - Random number generation
  - Stochastic models for options
  - Random walks
  - Improving performance

- Chapter 14—"Using C++ Libraries for Finance": Writing good financial code is not an individual task. You frequently have to use libraries created by other developers and integrate them into your own work. In the world of quantitative finance, a number of C++ libraries have been used with great success. This chapter reviews some of these libraries and explains how they can be integrated into your own derivative-based applications. Some of the topics covered include the following:
  - Standard library tools
  - QuantLib
  - Boost math
  - Boost lambda
- Appendix A—a quick summary of the changes introduced by C++20, for your reference.

Example Code

The examples given in this book have all been tested on MacOS X using the Xcode 7 IDE. The code uses only standard C++ techniques, so you should be able to build the given examples using any standards-compliant C++ compiler that implements the C++20 standard. For example, gcc is available on most platforms, and Microsoft Visual Studio will also work on Windows. The clang compiler is another option that is available in multiple platforms, including Windows, MacOS, and Linux.

If you use MacOS X and don't have Xcode installed in your computer yet, you can download it for free from the Apple store or from the Apple developer website at `http://developer.apple.com`.

If you instead prefer to use gcc on Windows, you can download the MinGW distribution from the website `www.mingw.org`.

Once MinGW is installed, start the command prompt from the MinGW program group in the Start menu. Then, you can type `gcc` to check that the compiler is properly installed.

To download the source code for all examples in this book, visit the web page of the author at `http://coliveira.net`, or navigate to `www.apress.com/9781484263143` and click the **Download Source Code** button.

CHAPTER 1

Options Concepts

In the last few decades, software development has become an integral part of the financial and investment industry. Advances in trading infrastructure, as well as the need for increased volume and liquidity, have prompted financial institutions to adopt computational techniques as part of their day-to-day operations. This means that there are many opportunities for computer scientists specializing in the design and development of automated strategies for trading and analyzing stocks, options, and other financial derivatives.

Options are among the several investment vehicles that are currently traded using automated methods, as you will learn in the following chapters. Given the mathematical structure and properties of options and related derivatives, it is possible to explore their features in a controlled way, which is ideal for the application of computational algorithms. In this book, I present many of the computational techniques currently used to develop strategies in order to trade options and other derivatives.

An *option* is a standard financial contract that derives its value from an underlying asset such as common stock, foreign currency, a basket of stocks, or a commodity. Options can be used to pursue multiple economic objectives, such as hedging against large variations on the underlying asset, or speculating on the future price of a stock. This chapter presents the basic concepts of options, along with supporting definitions. These concepts will be used in the next few chapters to describe algorithms and strategies with their implementation in C++20. In this chapter, I also give an overview of the use of C++ in the financial industry and how options can be modeled using this language.

C. Oliveira, *Options and Derivatives Programming in C++20*, https://doi.org/10.1007/978-1-4842-6315-0_1

The following concepts are explored in the next sections:

- *Basic definitions:* You will learn fundamental definitions about option contracts and how they are used in the investment industry.
- *Fundamental option strategies:* Due to their flexibility, options can be combined in a surprisingly large number of investment strategies. You will learn about some of the most common option strategies and how to model them using C++.
- *Option Greeks:* One of the advantages of options investing is that it promotes a very analytical view of financial decisions. Each option is defined by a set of mathematical quantities called *Greeks*, which reflect the properties of an option contracts at each moment in time.
- *Delta hedging:* One of the ways to use options is to create a hedge for some other underlying asset positions. This is called delta hedging, and it is widely used in the financial industry. You will see how this investment technique works and how it can be modeled using C++.

Basic Definitions

Let's start with an overview of concepts and programming problems presented by options in the financial industry. Options are specialized trading instruments and therefore require their users to be familiar with a number of details about their operation. In this section, I introduce some basic definitions about options and their associated ideas. Before starting, take a quick look at Table 1-1 for a summary of the most commonly used concepts. These concepts are defined in more detail in the remaining parts of this section.

Table 1-1. *Basic Concepts in Options Trading*

Concept	Definition
Call option	An option contracts that gives its owner the right to buy the underlying asset for a predetermined price during certain time period.
Put option	An option contracts that gives its owner the right to sell the underlying asset for a predetermined price during certain time period.
Underlying	Asset whose price is used as the base of the option contracts.
Strike price	The price at which option owners can buy or sell the underlying asset under the duration of the option contracts.
Expiration	The last date of validity for the option contracts.
Settlement	The act of liquidating the option contracts at the expiration date.
Intrinsic value	Amount of option value that is directly derived from the underlying price.
Time value	Amount of option value that is derived only from the time remaining in the option contracts.
Break-even price	The price at which an investor will start to make a profit in the option.
Exercise	The act of buying or selling the option underlying under the price determined by the option contracts.
American option	An option style where their owners can exercise the option contracts at any moment between option purchase and option expiration.
European option	An option style where option owners can exercise the option contracts only at expiration time.
ATM	(At the money): Term that refers to options that have a strike price close to the current price for the underlying.
OTM	(Out of the money): Term that refers to options that have a strike price above (for calls) or below (for puts) the current price of the underlying asset. These options have no intrinsic value.
ITM	(In the money): Term that refers to options that have a strike price below (for calls) or above (for puts) the current price of the underlying asset. These options have an intrinsic value.

Options can be classified according to several criteria. The features of these options determine every aspect of how they can be used, such as the quantity of underlying assets, the strike price, and the expiration, among others. There are two main types of option contracts: calls and puts. A *call* is a standard contract that gives its owner the right (but not the obligation) to buy an underlying instrument at a particular price. Similarly, a *put* is a standard contract that gives its owner the right (but not the obligation) to sell an underlying instrument at a predetermined price.

The *strike price* is the price at which the option can be exercised (i.e., the underlying can be bought or sold). For example, a call for IBM stock with strike $100 gives its owner the right to buy IBM stock at the price of $100. If the current price of IBM is greater than $100, the owner of such an option has the right to buy the stock at a price that is lower than the current price, which means that the call has a higher value as the value of IBM stock increases. This situation is exemplified in Figure 1-1. If the current price is lower than $100 at expiration, then the value of the option is zero, since there is no profit in exercising the contract. Clearly, the profit/loss calculation will depend on the price originally paid for the option and the final price at expiration.

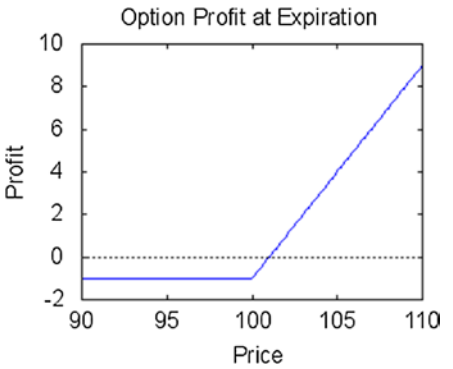

Figure 1-1. *Profit chart for a call option*

As you have seen in this example, if you buy a call option your possible gain is unlimited, while your losses are limited to the value originally paid. This is advantageous when you're trying to limit losses in a particular investment scenario. As long as you are okay with losing a limited amount of money paid for the option, you can profit from the unlimited upside potential of a call (if the underlying grows in price). Put options don't have unlimited profit potential since the maximum gain happens when the underlying price is zero. However, they still benefit from the well-defined, limited loss vs. the possible large gains that can be incurred.

Expiration: The expiration is the moment when the option contracts ends its validity and a final value exchange can be performed. Each option will have a particular, predefined expiration. For example, certain index-based options expire in the morning of the third Friday of the month. Most stock-based options expire in practice at the end of the third Friday of the month (although they will frequently list the Saturday as the formal expiration day). More recently, several weekly-based option contracts have been made available for some of the most traded stocks and indices. And finally, a few highly liquid trading instruments (such as S&P index funds) have expirations twice a week. Each option contracts makes it clear when expiration is due, and investors need to plan accordingly on what to do before expiration date.

Settlement: The settlement is the agreed-on result of the option transaction at expiration, the specific time when the option contracts expires. The particular details of settlement depend on the type of underlying asset. For example, options on common stock settle at expiration day, when the owner of the option needs to decide to sell (for puts) or buy (for calls) a certain quantity of stock. For index-based options, the settlement is normally performed directly in cash, determined as the cash equivalent for a certain number of units of the index. Some options on futures may require the settlement on the underlying commodity, such as grain, oil, or sugar. Investors need to be aware of the requirement settlement for different option contracts. Trading brokerages will typically let investors know about the steps required to settle the options they're currently holding.

Selling options: An investor can buy or sell a call option. When doing so, it is important to understand the difference between these two scenarios. For option buyers, the goal is to profit from the possible increase (in the case of calls) or decrease (in the case of puts) in value for the underlying. For option sellers, on the other hand, the goal is to profit from the lack of movement (increase for calls or decrease for puts). So, for example, if you sell calls against a stock, the intent is to profit in the case that the stock decreases in price or stays at a price lower than the strike price. If you sell a put option, the goal is to profit when the stock increases in price or stays higher than the strike price until expiration.

Option exercise: An option contracts can be used to buy or sell the underlying asset as dictated by the contract specification. This process of using the option to trade the underlying asset is called *option exercising*. If the option is a call, you can exercise it and buy the underlying asset at the specified price. If the option is a put, you can use the option to sell the underlying asset at the previously specified price. The price at which

the option is exercised is defined by the contract. For example, a call option for AAPL stock with a $100 strike allows its owner to buy the stock for the strike price, independent of the current price of AAPL.

Exercise style: Option contracts can have different exercise styles based on when exercising is allowed. There are two main types:

- *American options:* Can be exercised at any time until expiration. That is, the owner of the option can decide to exercise it at any moment, as long as the option has not expired.
- *European options:* Can be exercised only upon expiration date. This style is more common for contracts that are settled directly on cash, such as index-based options.

An option is defined as a derivative of an underlying instrument. The *underlying instrument* is the asset whose price is used as the basic value for an option contracts. There is no fixed restriction on the type of asset used as the underlying for an option contracts, but in practice options tend to be defined based on openly traded securities. Examples of securities that can be used as the underlying asset for commonly traded option contracts include the following:

- *Common stock:* Probably the most common way to use options is to trade call or put options on common stock. In this way, you can profit largely from price changes in stocks of well-known public companies such as Apple, IBM, Walmart, and Ford.
- *Indices:* An index, such as the Dow Industrials or the NASDAQ 100, can be used as the underlying for an option contracts. Options based on indices are traditionally settled on cash (as explained earlier), and each unit of value corresponds to multiples of the current index value.
- *Currencies:* A currency, usually traded using Forex platforms, can also be used as the underlying for option contracts. Common currency pairs involving the US dollar, euro, Japanese yen, and Swiss franc are traded 24 hours a day. The related options are traded on lots of currencies, which are defined according to the relative prices of the target currencies. Expiration varies similarly to stock options.

- *Commodities:* Options can also be written on commodities contracts. A commodity is a common product that can be traded in large quantities, including agricultural products such as corn, coffee, and sugar; fuels such as gasoline and crude oil; and even index-based underlying assets such as the S&P 500. Options can be used to trade such commodities and trading exchanges now offer options for many of the commodity types.
- *Futures:* These are contracts for the future delivery of a particular asset. Many of the commodity types discussed previously are traded using future contracts, including gasoline, crude oil, sugar, coffee, and other products. The structure of future contracts is defined to simplify the trade of products that will only be available within a due period, such as next fall, for example.
- *ETFs (exchange-traded funds) and ETN (exchange-traded notes):* More recently, an increasing number of funds have started to trade using the same rules applicable to common stocks in standard exchanges. Such funds are responsible for maintaining a basket of assets, and their shares are traded daily on exchanges. Examples of well-known ETFs include funds that hold components of the S&P 500, sectors of the economy, and even commodities and currency.

Options trading has traditionally been done on stock exchanges, just like other forms of stock and future trading. One of the most prominent options exchange is the Chicago Board Options Exchange (CBOE). Many other exchanges provide support and liquidity for the trading of options for many of the instruments listed here.

The techniques described in this book are useful for options with any of these underlying instruments. Therefore, you don't need to worry if the algorithms are applied to stock options or the futures options, as long as you consider the peculiarities of these different contracts, such as their expiration and exercise.

Options can also be classified according to the relation between the strike price and the price of the underlying asset. There are three cases that are typically considered:

- An option is said to be *out of the money* (OTM) when the strike price is above the current price of the underlying asset for call options, or when the strike price is below the current price of the underlying asset for put options.

- An option is said to be at the money (ATM) when the strike price is close to the current price of the underlying asset.
- An option is said to be in the money (ITM) when the strike price is below the current price of the underlying asset, for call options, or when the strike price is above the current price of the underlying asset, for put options.

Notice that OTM options are cheaper than a similar ATM option, since the OTM options (being away from the current price of the underlying) have a lower chance of profit than ATM options. Similarly, ATM options are cheaper than ITM options, because ATM options have less probability of making money than ITM options. Also notice that, when considering the relation between strike price and underlying price, the option price generally reflects the probability that the option will generate any profit.

A related concept is the *intrinsic value* of an option. The intrinsic value is the part of the value of an option that can be derived from the difference between strike price and the price of the underlying asset. For example, consider an ITM call option for a particular stock with a strike of $100. Assume that the current price for that stock is $102. Therefore, the price of the option must include the $2 difference between the strike and the price of the underlying, since the holder of a call option can exercise it and have an immediate value of $2. Similarly, ITM put options have intrinsic value when the current price of the underlying is below the strike price, using the same reasoning.

The *break-even price* is the price of the underlying on expiration at which the owner of an option will start to make a profit. The break-even price has to include not only the potential profit derived from an increase in intrinsic value but also the cost paid for the option. Therefore, for an investor to make a profit on a call option at expiration, the price of the underlying asset has to rise above the strike plus any cost paid for the option (and similarly it has to drop below the strike minus the option cost for put options). For example, if an $100 MSFT call option has a cost of $1, then the investor will have a net profit at expiration only when the price of MSFT rises above $101 (and this without considering transaction costs).

As part of the larger picture of investing, options have assumed an important role due to their flexibility and their profit potential. As a result, new programming problems introduced by the use of options and related derivatives have come to the forefront of the investment industry, including banks, hedge funds, and other financial institutions. As you will see in the next section, C++ is the ideal language to create efficient and elegant solutions to the programming problems occurring with options- and derivatives-based investing.